REMARKS

Claims 1-15 are now pending in this application. Claims 1-15 have been rejected under the judicially created doctrine of obviousness-type double patenting as being unpatentable over Claims 1-14 of U.S. Patent No. 6, 052,707 to D'Souza ("D'Souza"). Claims 12-13 have been rejected under 35 U.S.C. § 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which the applicant regards as the invention. Claims 1-4, and 11-13 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over cited references to the article Windows Multitasking by Jeff Prosise ("Prosise"), the article Windows Internals by Matt. Pietrek ("Pietrek"), and the book Running Windows 3.1 by Craig Stinson ("Stinson"). Claims 5-7 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Prosise, Pietrek, and Stinson as applied to Claim 1, and further in view of U.S. Patent No. 4,980,824, issued Dec. 25, 1990 to Tulpule et al. ("Tulpule"). Although not stated with clarity, it appears that Claim 14 also stands rejected, similarly to Claim 5, under 35 U.S.C. § 103(a) as being unpatentable over Prosise and Pietrek, in view of Tulpule.

As D'Souza is commonly assigned to the assignee of the present application, a terminal disclaimer in compliance with 37 C.F.R. § 1.321(c) is being concurrently filed herewith to overcome the rejection of Claims 1-15 based on the non-statutory double patenting ground. Claim 12 has been amended to more particularly point out and distinctly claim the subject matter which the applicant regards as the invention to overcome the rejection of Claims 12-13 under 35 U.S.C. § 112, second paragraph. Pursuant to 37 C.F.R. § 1.111, and for the reasons set forth below, the applicant respectfully requests reconsideration and allowance of this application.

Summary of the Invention:

The present invention combines cooperative or non-preemptive scheduling with preemptive scheduling to optimize the scheduling of tasks by an operating system. The difficulty encountered with non-preemptive scheduling is that it requires cooperation between tasks and sometimes such cooperation cannot be achieved. For example, when the processor performs a task, the task may allocate a resource. If the task hangs up while it has control of the resource, the task cannot release the resource to allow the processor to perform the next task. Moreover, one task may monopolize the processor to the detriment of other tasks that are awaiting processor time. Preemptive scheduling avoids these drawbacks by ensuring that each task only has control of the processor or a particular resource for a predefined period of time.

The present invention strikes a compromise between non-preemptive and preemptive scheduling models. The present invention provides non-preemptive scheduling to provide compatibility with applications designed for specific operating systems, such as those designed to run on the Microsoft Windows, version 3.1, operating system, owned and licensed by the Microsoft Corporation of Redmond, Washington. Interdependent tasks are logically partitioned into groups. Thus, each group holds a number of interdependent tasks. These interdependent tasks are non-preemptively scheduled. The groups, however, are preemptively scheduled such that each group is given a time slot in which the group's non-preemptively scheduled tasks may execute. By logically partitioning the tasks into groups of interdependent tasks, the present invention avoids the data sharing and other problems that may arise with a strictly preemptive scheduling approach. Each task may initially start its own group. As tasks execute and interdependencies are determined, the tasks are grouped according to their interdependencies, including moving the task into a group with other applications which use the same resource to avoid resource sharing problems. Each task includes a task dependency list that is generated by performing a union of all of the modules in the module dependency list of the task.

Summary of Principal References Cited

Prosise describes the use of non-preemptive scheduling of Windows applications in a System Virtual Machine ("VM") and preemptively scheduling that System VM with other DOS VMs. Prosise does not describe, teach, or suggest grouping the Windows applications within the System VM based on interdependencies between the Windows applications. Moreover, Prosise makes no suggestion that non-preemptive scheduling may be used within a DOS VM.

The Office Action suggests that the use of non-preemptive scheduling within the DOS VMs would be obvious. However, a closer examination of the Prosise reference reveals that such is not the case. At the top of the first column of page 263, Prosise explains that cooperative processing occurs through the message queue in use by each Windows application. As discussed at the bottom of the third column on page 261, a Windows application retains control of processing so long as it has messages to process in the system message queue. When it retrieves a "WM_QUIT" message from the message queue, the Windows application relinquishes control of processing to the next Windows application. In contrast, a DOS application running in a DOS VM does not interact with the system message queue. It would not have been obvious to add cooperative scheduling to the DOS VMs because the DOS applications do not interact with the system message queue in a manner that would enable relinquishing control of processing to other tasks. Adding cooperative scheduling to the DOS VMs would in no way achieve the simplicity and consistency suggested in the Office Action.

Pietrek discusses, *inter alia*, the concepts of modules and tasks in the Windows operating system. Pietrek mentions the interdependencies of tasks based on shared modules as one motivator for non-preemptive multitasking. Pietrek, pg. 215, third paragraph (four copies of CALC.EXE may be simultaneously executed and the system uses the module table to coordinate the sharing). Pietrek describes, in very general terms, non-preemptive or cooperative multitasking. Nothing in Pietrek describes, teaches, or suggests the grouping of tasks based on their interdependencies so that those groups may be preemptively multitasked.

Tulpule discloses event-driven execution ordering of tasks on one or more signal processors based on dependencies and pre-requisites previously established for each task using a precedence graph. Nothing in Tulpule describes, teaches, or suggests the grouping of tasks based on their interdependencies so that those groups may be preemptively multitasked.

Comment on Dates of References:

As in the examination of the parent application of the present application, the Office Action asserts that Prosise and Pietrek discuss features of Windows 3.1, and states that Stinson is cited to show that Windows 3.1 was available in 1992. Based on that, the Office Action concluded that the teachings of Prosise and Pietrek were available in 1992. As during the examination of the parent application, the applicant respectfully disagrees. The applicant admits that Stinson shows that the Windows 3.1 operating system was available in 1992. However, the Prosise reference discusses multiple operating systems, including Windows 3.1, Windows NT, OS/2, and "Windows 3.*x*." It is a common convention to use a small italicized "x" after the decimal point in a version identifier to indicate multiple versions of a software program.

The Office Action indicates that version 3.1 of the Windows operating system was available as early as 1992. While that may be true, multiple versions of the Windows operating system were subsequently available. For instance, a Windows 3.11 was available at some time later than 1992. There is no way to determine with accuracy from Prosise which of the several versions of the Windows operating system was intended by the version indicator "3.*x*" recited in that reference. For that reason, while the applicant admits that portions of the discussion in Prosise probably describe technology available as early as 1992, the applicant strongly disagrees with the Office Action's conclusion that all of the teachings of the Prosise reference were necessarily available as early as 1992. Likewise, the applicant disagrees with the Office Action's conclusion that all the teachings of Pietrek were necessarily available at any time prior to the publication date of that reference. The references simply do not support that conclusion.

The Claims Distinguished

The present invention employs a combination of preemptive and non-preemptive multitasking not taught or suggested by the prior art. Interdependent tasks are grouped and scheduled non-preemptively, whereas the groups of interdependent tasks are scheduled preemptively. Among other reasons, the interdependency of two tasks is based on whether those two tasks allocate a common resource or call a common module of code. Moreover, the grouping of the interdependent tasks is updated in response to a change in the interdependencies of the tasks, i.e., a change in the allocation of common resources, or in the calling of a common module of code.

Rejection of Claims 1-4 and 11-13 Under 35 U.S.C. § 103:

Claims 1-4, and 11-13 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Prosise, Pietrek, and Stinson.

As to Claims 1, 11, and 12, the Office Action asserts that Prosise teaches tasks, logically partitioning tasks into groups of interdependent tasks, preemptively scheduling the groups, each group is given a time slot, non-preemptively scheduling within a group. The Office Action notes that Windows OS and DOS operating systems each provides resources in the form of system routines to be utilized/called by application programs, and that the Windows applications grouped under System VM use Windows resources, and DOS applications grouped under DOS VMs call/utilize DOS resources. The Office Action concedes that Prosise does not teach that the non-preemptively scheduling is performed within *each* of all the groups (emphasis in Office Action), but that to do so would have been an obvious choice to provide simplicity and consistency. The applicant respectfully disagrees. As noted above, adding cooperative scheduling to the DOS VMs would in no way achieve the simplicity and consistency suggested in the Office Action. DOS applications simply do not interact with the system message queue in a manner that would enable relinquishing control of processing to other tasks. Thus, it would not have been obvious, or even advantageous to add cooperative scheduling to the DOS VMs.

As to Claim 2, the Office Action asserts that Pietrek teaches storing a group list for each associated group. The Office Action equates Pietrek's task database (TDB) with a group list in that the TDB is apparently maintained in a linked list. The Office Action further asserts that Pietrek teaches identifying information for tasks because each TDB contains a selector that apparently identifies the next TDB, i.e. the linked list. The applicant respectfully disagrees. The TDB is not a group list as recited in Claim 2, but rather a series of tasks that happen to be linked together through the use of the selectors. Moreover, Pietrek does not disclose identifying information for tasks included in the associated group as recited in Claim 2. The applicant notes that the selectors of Pietrek are not information that is identified for the tasks, but merely identifiers of the tasks themselves.

As to claim 3, the Office Action asserts that the TDB of Pietrek teaches storing status information, whether the group has a task that is running, and holding identifying information about any task that is running. The applicant respectfully disagrees. The information revealed in the format of the TDB disclosed in Pietrek is simply information related to a task, and not to a group as recited in Claim 3.

As to claim 4, the Office Action asserts that the Windows application disclosed in Prosise "as modified" teaches initially placing each task in its own group, apparently equating the Windows application with a task in a group. Note that the applicant requests clarification of what other reference modifies Prosise in such a way as to teach initially placing each task in its own group as asserted in the Office Action. The Office Action further asserts that the Windows applications under System VM disclosed in Prosise in Figure 1 teaches subsequently combining groups of tasks into a merged group, and that the shared resources disclosed in Pietrek teaches that each task in the merged group calls a common resource. The applicant respectfully disagrees. Neither Prosise nor Pietrek disclose a process of logically partitioning tasks into groups of tasks that utilize the same resources, much less doing so by initially placing each task in its own group, and subsequently combining groups of tasks into a merged group wherein each task in the merged group allocates a common resource when run.

Rejection of Claims 5-7, and 14 Under 35 U.S.C. § 103:

Claims 5-7 and 14 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Prosise, Pietrek, and Stinson as applied to Claim 1, and further in view of U.S. Patent No. 4,980,824, issued Dec. 25, 1990 to Tulpule et al. ("Tulpule").

As to claim 5, the Office Action asserts that Pietrek teaches modules of code and providing a task dependency list for each task that lists resources that are candidates to be allocated when the task is run, apparently equating the TDB of Pietrek with the task dependency list recited in Claim 5, noting in particular that Pietrek discloses the creation of a segment that contains the module handles of all the DLLs that a loading executable module brought into memory. The applicant respectfully disagrees. The segment disclosed in Pietrek may identify whatever DLL's that a particular executable happens to be using in a particular instance, but that is not the same as listing the resources that are candidates to be allocated when the task is run on the processor as recited in Claim 5.

Also, as to Claim 5, the Office Action further asserts that Prosise teaches that interdependent tasks, such as Windows applications, are grouped together, such as when Windows applications are grouped to be run under System VM. The Office Action further asserts that Prosise teaches that modules run by the tasks, such as Windows resources and Windows DLLs, are included in a single group, i.e. under a Windows group. The examiner argues that since Pietrek teaches describing modules related to a task with a task dependency list, that the combined teaching of Prosise and Pietrek discloses task dependency lists for the Windows and DOS applications. The applicant respectfully disagrees. The disclosure of Prosise of grouping Windows applications to run under System VM does not teach or disclose the grouping of interdependent tasks. That modules run by Windows applications, such as Windows resources and DLLs, could be considered part of a Windows group is not the same as interdependent tasks that are grouped together because of their interdependency as recited in Claim 5.

Also, as to Claim 5, the Office Action further asserts that Tulpule teaches partitioning tasks based on interdependencies, including examining task dependency lists, and that one of ordinary skill in the art would have been motivated to combine the teachings of Prosise and Pietrek with Tulpule because this would have provided a flexible configuration to dynamically respond to changes of task's execution times. The applicant respectfully disagrees. As noted above, Tulpule discloses event-driven execution ordering of tasks on one or more signal processors based on dependencies and pre-requisites previously established for each task using a precedence graph. Nothing in Tulpule describes, teaches, or suggests the grouping of tasks based on their interdependencies so that those groups may be preemptively multitasked. Thus, nothing in Tulpule, Prosise, and Pietrek would have motivated one of ordinary skill in the art to combine their respective teachings.

As to Claims 6-7, the Office Action asserts that they are rejected for the same reasons as Claims 2-3. The applicant respectfully disagrees. The TDB is not a group list as recited in Claim 6, but rather a series of tasks that happen to be linked together through the use of the selectors. Moreover, Pietrek does not disclose identifying information for tasks included in the associated group as recited in Claim 6. The applicant again notes that the selectors of Pietrek are not information that is identified for the tasks, but merely identifiers of the tasks themselves. In addition, the information revealed in the format of the TDB disclosed in Pietrek is simply information related to a task, and not to a group as recited in Claim 7.

As to Claim 14, the Office Action asserts that Claim 14 is rejected for some of the same reasons as Claim 5. The applicant respectfully disagrees. The disclosure of Prosise of grouping Windows applications to run under System VM does not teach or disclose the grouping of interdependent tasks. That modules run by Windows applications, such as Windows resources and DLLs, could be considered part of a Windows group is not the same as interdependent tasks that are grouped together because of their interdependency as recited in Claim 14.

Amendment to Claim 12 under 35 U.S.C. § 112:

Claim 12 has been amended to more particularly point out and distinctly claim the subject matter which the applicant regards as the invention. In pertinent part, Claim 12, as currently amended, now recites "logically partitioning tasks into groups of interdependent tasks, wherein said tasks are to be run non-preemptively," to clarify that it is the tasks that are to be run non-preemptively, and not the groups. The applicant submits that Claim 12, as currently amended, and dependent Claim 13, are in condition for allowance.

CONCLUSION

In view of the foregoing, it is submitted that the present application is now in condition for allowance. Reconsideration and re-examination of the application, as amended, and allowance of the claims at an early date is solicited. If the Examiner has any questions or comments concerning this matter, the Examiner is invited to contact the applicant's undersigned attorney at the number below.

Respectfully submitted,

CHRISTENSEN O'CONNOR
JOHNSON KINDNESS^PLLC^

Donna J. Coningsby

Donna J. Coningsby
Registration No. 41,684
Direct Dial No. 206.695.1719

I hereby certify that this correspondence is being deposited with the U.S. Postal Service in a sealed envelope as first class mail with postage thereon fully prepaid and addressed to Mail Stop Amendment, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the below date.

Date: June 25, 2004 Patricia Gubble

DJC:pag

LAW OFFICES OF
CHRISTENSEN O'CONNOR JOHNSON KINDNESS^PLLC^
1420 Fifth Avenue
Suite 2800
Seattle, Washington 98101
206.682.8100